Peer-to-peer warehousing of XML resources

Serge Abiteboul * — **Ioana Manolescu** * — **Nicoleta Preda** *

* *INRIA Futurs & LRI PCRI, FRANCE - firstname.lastname@inria.fr*

ABSTRACT. *We present* KADOP*, a distributed infrastructure for warehousing XML resources in a peer-to-peer framework.* KADOP *allows users to build a shared, distributed repository of resources such as XML documents, semantic information about such documents, Web services, and collections of such items.* KADOP *builds on distributed hash tables as a peer communication layer, and ActiveXML as a model for constructing and querying the resources in the peer network. We describe* KADOP*'s data model, query language, and query processing paradigm.*

RÉSUMÉ. *Nous présentons KadoP, une infrastructure distribuée pour entreposer des ressources XML dans un environement pair-à-pair. KadoP permet aux utilisateurs de construire un entreôt partagé et distribué de ressources telles que des documents XML, des informations sémantiques sur ces documents, des services Web et des collections de tels objets. KadoP utilise des tables de hachage pour la communication entre les pairs et ActiveXML comme modèle pour construire et interroger les ressources du réseau de pairs. Nous décrivons le modèle de données de KadoP, le langage d'interrogation et le paradigme de traitement des requêtes.*

KEYWORDS: *XML warehousing, peer to peer, semantic web*

MOTS-CLÉS : *entrepôt de données XML, pair-à-pair, web sémantique*

1. This work is partially funded by the French government research grant ACI MDP2P.

1. Introduction

The increasing popularity of P2P architectures and Web services as a data exchange mechanism opens up new possibilities for building very large-scale data management applications. We demonstrate KADOP[1], a system for constructing and maintaining, in a decentralized, P2P style, a warehouse of *resources*. KADOP allows a user to perform the following tasks:

(i) *publish* XML resources, making them available to all peers in the P2P network and in particular maintain indexing up to date; (ii) *search* for resources meeting certain criteria (based on content, structure as well as semantics of the data); (iii) *declaratively build thematic portals* from resources of the system.

KADOP leverages several existing technologies and models such as distributed hash tables (DHT), Active XML documents, query processing using structural joins.

This document is structured as follows. Section 2 describes the KADOP data model, and Section 3 its query laguage. We present the system architecture and demonstration scenario in Section 4, and discuss related works in Section 5.

2. KADOP data model

KADOP's data model is generic, and focused on the types of resources that can be published in our system (Figure 1). We distinguish two kinds of resources: *data items*, and *semantic items*. **Data items** (at bottom in Figure 1) model resources with various granularities: A *page* is an XML document; A *page fragment* is a subtree of a page and a *collection* is a user-defined set of data items; Inside pages, we also consider element *labels*, *attribute names*, and *words*. A *web service* is a function taking as input typed XML fragments, and returning a typed XML fragment. Data items are connected by *partOf* relationships. **Semantic items** depicted in the upper part of Figure 1, consist of *concepts* connected by two types of relationships: *partOf*, and *narrowerThan*.

Any item is uniquely identified by the PID (peer ID) of the host and a name within the peer. A data item may be connected to a concept via an *relatedTo* relationship, reflecting the fact that the data item is pertinent to the concept. Such relationships can be (i) specified by a user; (ii) infered automatically between elements matching a DTD type τ, and the corresponding concept c_τ with the same name; (iii) derived automatically by a document classifier.

Example Figure 2 shows a sample instance of the KADOP internal data model, over two peers. A page is depicted as a tree; next to the root, we show the PID and the page name. Rounded boxes contain concepts and relationships between them. Collections, and collection memberships, are listed in italic font, and *relatedTo* statements appear in diamond-shaped boxes.

3. KADOP query language

The KADOP query language allows retrieving *data items*, based on conditions on these items, and on their relationships with various concepts. A KADOP query Q is a tree pattern, whose nodes represent data items, and whose edges represent containment relationships among the nodes. Each node may be annotated with: (i) *a name condition*, such as a tag, document, or collection name, or a $*$ matching any name;

1. KADOP stands for: ***K**nowledge **a**nd **d**ata **o**n a **P**2P network.*

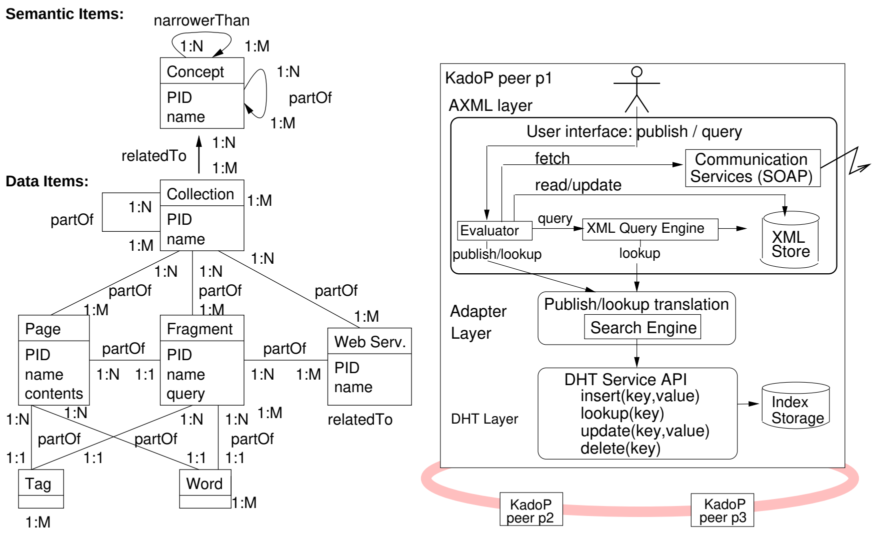

Figure 1. *KADOP's data model (left), system architecture and functionalities (right).*

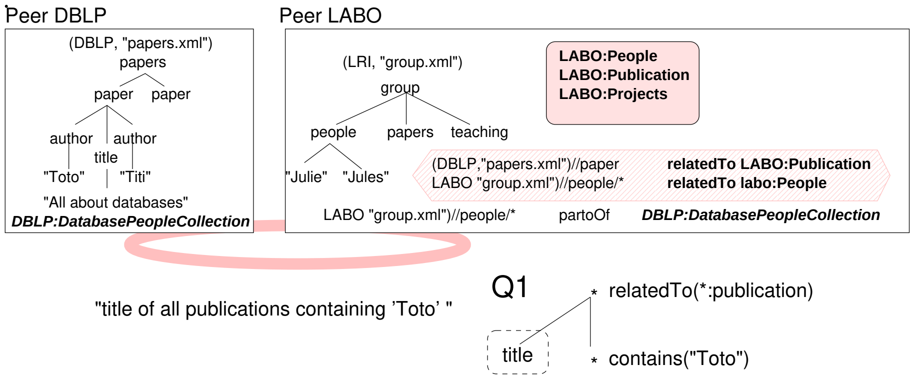

Figure 2. *Instance of the* KADOP *data model (upper), query sample (lower).*

(*ii*) *semantic conditions* of the form *relatedTo* c, where c denotes a concept, using a name, and either a PID or a $*$; and (*iii*) *textual conditions* of the form $contains\ \ w$, where w is a word. We distinguish a single *return node* N_R of Q.

Example Figure 2 shows sample of of KADOP query; query nodes are labeled by their names, and the returned node is shown in a dashed-line box. On the configuration in Figure 2, Q_1 returns the titles of DBLP papers containing "Toto", since they are instances of LABO:publication.

4. System architecture and demo scenarios

A KADOP peer is built in several layers (Figure 1), some of which use AXML [HTT] documents. An AXML document is a *partially intensional* XML document, containing calls to Web services; when a call is activated, the result is inserted in the document. The DHT provides physical resource indexing services. These are provided as a distributed hash table, in which key-value pairs can be registered using the *put(k,v)* method, while key lookup can be performed using *get(k)*.

Users publish resources by activating calls to a *publishing Web service* provided by the local peer. This service writes the resources in the local XML Store, and calls the *put* Web service of the DHT layer to distribute the index over the DHT to enable all peers to look up them. To answer a user query, search keys are extracted from each query node, and a call to the *get* DHT Web service is issued for each key. The calls return element IDs in the KADOP network, which are combined through structural joins as in [ALK 02], to identify the pages matching the query. Communication Services are called to fetch the useful parts of these pages that are stored locally.

The crux of this architecture lies in the choice of the *key-value* pairs to insert in the DHT index. The idea is that *search criteria* (such as category or tag names, precise words etc.) make up keys, while *resource location* (PID and precise location of a resource in the peer) make up the associated values.

Demonstration scenario We demonstrate several functionalities of KADOP, based on the large, heterogeneous INEX [INE04] corpus of computer science-related XML data; KADOP participates to INEX this year. **Publishing and indexing** We demonstrate resource publishing, and (for illustration) show the P2P indexing entries computed by KADOP. We show how KADOP leverages the intensional aspect of AXML (documents may include service calls), to intensionally index resources. The value associated to the key is not the location of a resource, but of a Web service that, when called, will return pertinent resources. Intensional indexing is much more concise: a single intensional index entry replaces several extensional (regular) entries, at the expense of one extra service call. **Querying and portal construction** We demonstrate KADOP query formulation and sequence of processing steps. Finally, we show how a portal on a given topic can be built as an AXML document. Such a portal is *intensional*, meaning its actual data items are only retrieved on demand, by activating a service call execution in the P2P network.

5. Status and related works

We are currently implementing KADOP using Pastry [Fre] and AXML [HTT]; other efficient P2P structures could be used. Our system is related to the existent XML-based P2P frameworks and semantic P2P networks but it improves by using a DHT layer allowing each peer to search for resources anywhere in the P2P network. More motivations for our work may be found in [ABI 03].

6. References

[ABI 03] ABITEBOUL S., "Managing an XML Warehouse in a P2P Context", *Int'l CAISE Conf.*, 2003.

[ALK 02] AL-KHALIFA S., JAGADISH H., PATEL J., WU Y., KOUDAS N., SRIVASTAVA D., "Structural Joins: A Primitive for Efficient XML Query Pattern Matching", *Proc. of Int. Conf. on Data Engineering (ICDE)*, 2002.

[Fre] "FreePastry", www.cs.rice.edu/CS/Systems/Pastry/FreePastry/.

[HTT] HTTP://ACTIVEXML.NET/, "ActiveXML home page", http://purl.org/net/axml.

[INE04] "*IN*itiative for the *E*valuation of *X*ML retrieval", inex.is.informatik.uni-duisburg.de:2004, 2004.